DOI: 10.1111/cgf.15099
Eurographics Conference on Visualization (EuroVis) 2024
W. Aigner, D. Archambault, and R. Bujack
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 43 (2024), Number 3

AutoVizuA11y: A Tool to Automate Screen Reader Accessibility in Charts

Diogo Duarte[1], Rita Costa[1], Pedro Bizarro[1], and Carlos Duarte[2]

[1]Feedzai, Portugal
[2]LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal

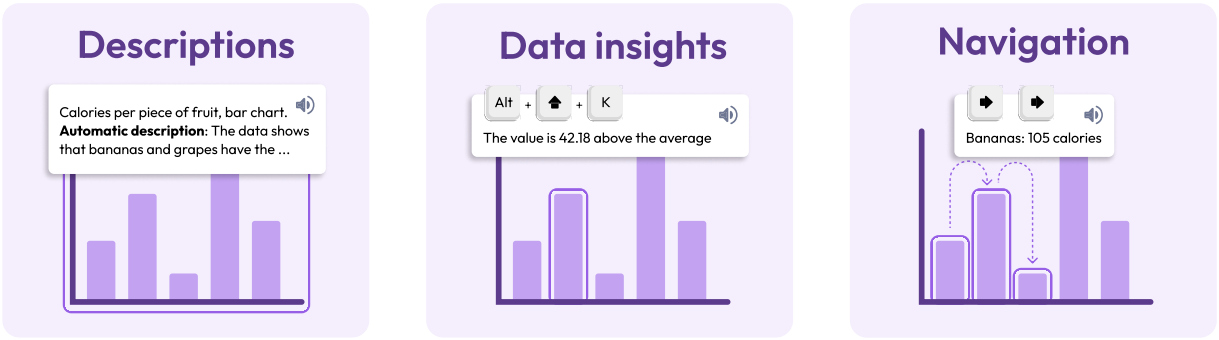

Figure 1: *AutoVizuA11y is a tool designed to automate the creation of accessible charts for users of screen readers. It automatically generates descriptions of the data, calculates insights about the data, and provides keyboard navigation between charts and underlying elements. All of these features are supported by a wide range of keyboard shortcuts.*

Abstract

Charts remain widely inaccessible on the web for users of assistive technologies like screen readers. This is, in part, due to data visualization experts still lacking the experience, knowledge, and time to consistently implement accessible charts. As a result, screen reader users are prevented from accessing information and are forced to resort to tabular alternatives (if available), limiting the insights that they can gather. We worked with both groups to develop AutoVizuA11y, a tool that automates the addition of accessible features to web-based charts. It generates human-like descriptions of the data using a large language model, calculates statistical insights from the data, and provides keyboard navigation between multiple charts and underlying elements. Fifteen screen reader users interacted with charts made accessible with AutoVizuA11y in a usability test, thirteen of which praised the tool for its intuitive design, short learning curve, and rich information. On average, they took 66 seconds to complete each of the eight analytical tasks presented and achieved a success rate of 89%. Through a SUS questionnaire, the participants gave AutoVizuA11y an "Excellent" score — 83.5/100 points. We also gathered feedback from two data visualization experts who used the tool. They praised the tool availability, ease of use and functionalities, and provided feedback to add AutoVizuA11y support for other technologies in the future.

CCS Concepts

- ***Human-centered computing → Visualization; Accessibility***

1. Introduction

It is estimated that 33.6 million people suffer from blindness and 206 million from moderate and severe vision impairment [Bli20]. Many of these individuals rely on assistive tools to perform daily tasks in both physical and digital environments. Notably, in the context of web browsing, screen readers (SR) stand out as the most commonly used assistive technology due to their widespread availability and reduced cost [KJRK22]. The American Foundation for

the Blind defines screen readers as "software programs that allow blind or visually impaired users to read the text that is displayed on the computer screen with a speech synthesizer or braille display" [AFP24]. However, this technology alone does not mean that blind users have experiences similar to those of their sighted peers when navigating the web. A 2023 report from WebAIM revealed that 96.3% of the top one million websites failed to meet Web Content Accessibility Guidelines (WCAG) 2.1 standards [Web21].

When it comes to accessing data visualizations, it has been shown that SR users extract information 61% less accurately and spend 211% more time interacting with charts on the web than non-SR users [SCWR21]. Through a series of interviews with five SR users we identified that these difficulties are often overcome by seeking alternative data exploration methods (when available), such as interacting with HTML tables, exporting the data to spreadsheet analysis software, or seeking assistance from sighted peers. The barriers associated with chart inaccessibility can jeopardize access to information, alienate people, and even prevent individuals who rely on screen readers from accessing certain jobs that they would otherwise not struggle to get. However, it's important to note that despite these challenges, our evaluations have shown that SR users, when provided with accessibly built charts, are able to successfully explore data visualizations within a short time frame.

Despite recent significant developments, with the introduction of tools that aim to answer some of the challenges that SR users face when coming across visualizations on the web, there is still room for improvement. Mostly because those tools still lack in chart coverage and diversity of features that allow SR users to quickly explore and understand varied sets of charts. Following requirements from literature and a formative study conducted with both daily SR users and Data Visualization (DV) experts, we created AutoVizuA11y. Built upon the heuristics outlined in Chartability [EBM22] and WCAG [W3C19], the tool improves user experience by supporting three fundamental sets of features: rich descriptions, data insights, and enhanced navigation.

Generating insightful descriptions that accurately identify trends is something existing tools struggle to automatically provide [SF18, SWM*22b, ZT15, EMW19]. AutoVizuA11y can optionally output descriptions of visualization's data using a Large language model (LLM) accessed trough the OpenAI API. Therefore, it eliminates the need for manually writing descriptions, even though it is still possible to do so. It works both with simple charts (such as bars, lines, and pies) and more complex ones (like scatterplots, heatmaps, and multi-series lines). It was designed to support the creation of charts using any low-level visualization library in React, like visx [Air17] or D3 [MB11], where charts and data are represented in the DOM using SVG elements. Additionally, AutoVizuA11y facilitates keyboard navigation in the underlying data of the chart. It also allows to navigate between charts, in case more than one is present on the page (useful for dashboards and infographics). It automatically adds informative labels to each data element and offers an extensive array of keyboard shortcuts, thereby contributing to accelerating the exploration, facilitating data insight retrieval, and enabling the comparison of specific points against insights. To work, AutoVizuA11y requires minimal input from DV experts (accessibility expertise is not a requirement).

Following a user-centered design methodology, a usability study was conducted using AutoVizuA11y. A group of 15 SR users was asked to explore an interface containing charts created using AutoVizuA11y. The majority of the participants showed great interest in interacting with visualizations that incorporate AutoVizuA11y's features and stated a preference over commonly used chart alternatives like tables or data exports.

The main contributions of our work are:

1. We identified a set of challenges from SR users and DV experts regarding inaccessible charts on the web. These challenges emerged from ten interviews and led to the design requirements that guided the development of AutoVizuA11y;
2. We designed and implemented AutoVizuA11y, a tool that enables data visualizations to be fully explored by SR users. It integrates enhanced features of previously introduced tools, such as keyboard navigation, shortcuts, and statistical insights, with automatic data descriptions generation, without chart/data type restrictions. AutoVizuA11y does not rely on the DV experts' accessibility knowledge and only requires already available information about the chart. We made it open-source on `https://github.com/feedzai/AutoVizuA11y`;
3. Motivated by measuring the performance of the tool, we collected qualitative and quantitative results from a usability test where 15 SR users had to complete tasks in an interface with multiple AutoVizuA11y charts. The participants were able to complete each task, on average, in 66 seconds with a success rate of 89%.

2. Related work

2.1. Accessibility Guidelines

Over the past three decades, governments and civil society have developed plans to establish and enforce accessibility regulations. For software, WCAG [W3C19] is the reference framework. It consists of a set of guidelines designed to ensure digital content is accessible to people with disabilities. But it lacks specific recommendations for data visualization [EBM22, SB22, WPA*21].

To address such limitation, a group of experts extended it by creating Chartability [EBM22], a collection of heuristics aimed at improving chart accessibility. These serve as test cases during a data visualization audit to identify potential design failures. They relate to different disability barriers, and are not exclusive to SR users. The heuristics relevant to SR users were taken into account while creating AutoVizuA11y.

2.2. Accessibility in Data Visualization

In the Urban Institute's "Do No Harm Guide" from December 2022, focused on accessibility in Data Visualization [SPF22], experts claimed that researchers overly concentrate on color-related issues when creating accessible charts for the web while other areas of accessibility research are less explored — such as the usage of assistive technology like screen readers. Nevertheless, there has been a growing interest by the data visualization research community [KJRK21] with several authors shedding light on the habits and challenges these users face.

A recent study [SCWR21] revealed that SR users are less accurate and spend more time interacting with online data visualizations compared to non-SR users. This is a consequence of charts not being created with accessibility in mind, ending up with an improper code hierarchy, incorrect HTML elements usage, and not using ARIA properties correctly [KJRK22].

But if charts were accessible, what information would users like to know? In a study from Jung et al. [JMK*21], most blind and visually impaired individuals said they value knowing the data points and chart axis; close to half highlighted data trends and the type of charts; while information about color remained divisive.

Regarding how the information should be presented to the user, Kim et al. [KJRK21] proposed a model for SR users to access chart information, which includes notifying its existence, providing an overview of the visualization, and giving details about the data and context when requested. Sharif et al. [SCWR21] also found in their interviews that charts should be discoverable, provide an holistic data overview, and allow for the exploration of labeled data. According to them, additional exploration modes such as tables, sonification, and textual descriptions, should also be added. However, representing data with data in tables or through sound also poses its own challenges, as shown by Wang et al [WWJK22].

One critical heuristic of Chartability emphasizes the importance of providing a title, summary, context, or caption for charts [BBK*15], ultimately creating a description of the data. These should use simple language, have between two and eight sentences, and present the chart type before the summary [JMK*21]. Lundgard et al.'s four-level model of semantic content [LS21] guides the creation of effective chart descriptions, encompassing various levels of information and complexity. The first level describes fundamental chart characteristics while the forth (and last) provides contextual and domain-specific knowledge.

Currently available tools mostly address levels 1 and 2 [SF18, SWM*22b, ZT15, EMW19], which can be derived from the chart's data. Because levels 3 and 4 depend on perception and cognitive abilities, their automatic generation present a technological challenge [SPF22]. Evaluations conducted by the model authors [LS21] show that SR users prefer levels 2 and 3.

Informed by insights from previous authors, AutoVizuA11y provides a comprehensive range of information to SR users, delivered through multiple channels that offer both an holistic overview and selective exploration of data and statistical insights.

2.3. Accessibility tooling

Adding accessibility to data visualizations can be achieved in one of three ways: by using a charting library that already has some sort of accessibility considerations; adding the accessibility features by hand to a chart; or using a package independent from the visualization that will add some accessibility features. With current tooling, all these approaches have potential for improvement.

Highcharts [Hig09] and EvoGraphs [SF18] are visualization libraries that offer some accessibility features for SR users. They lack automatic detailed data descriptions, provide no statistical insights, have limiting customization and chart types for DV experts.

Chart.js [Dow13], D3 [MB11], and visx [Air17] are also visualization libraries, highly customizable and flexible. But they do not have accessibility considerations by default.

Then, there are the tools that approach this problem from a different perspective. Instead of offering methods to create data visualizations, they propose a top layer that adds the accessibility features. These tools vary in complexity, information provided, and modality. This is the approach taken with AutoVizuA11y.

Voxlens [SWM*22b] supports single and multi-series charts built with D3, Google Charts, and ChartJS. It provides a question-and-answer mode, a holistic summary of the data contained in the visualization and a sonified version of the chart, using the Sonifier [SWM22a] library. It requires very few inputs from the creator but, in contrast, does not label data elements inside the chart or allow users to navigate between them.

Another similar tool is Olli [BZS22]. It supports charts built with Vega, Vega-Lite, and Observable Plot but can be extended to support other Javascript visualization libraries. This tool creates a tree-like navigation structure between the different chart elements. But it only provides a basic description of the chart, with no way of accessing statistical insights through a summary or shortcuts.

The recently introduced Data Navigator [ENM23] enables the navigation of charts using multiple modalities beyond a keyboard and does not restrict the DV experts to a specific chart type or visualization library. Even though these modalities tailor the charts to different assistive technologies other than SRs, this tool only covers the navigation aspect of chart accessibility.

None of the explored approaches supports a portfolio of features adaptable to various visualization libraries and chart types. Furthermore, none of these approaches automatically incorporates all necessary accessibility features for SR users without relying on the DV experts' time, knowledge, and effort. Additionally, none automatically produces a data summary beyond level 2 complexity. AutoVizuA11y combines features that build upon the tools mentioned above and expands on them by allowing navigation between charts, shortcuts that compare data points and provide rich insights, and automating the generation of more complex data descriptions.

2.4. LLMs support for DataViz

LLMs, a type of Artificial Intelligence (AI) model based on the transformer architecture, have gained significant attention for their ability to process data and generate text resembling that produced by humans [OH20]. They can be applied to content generation, translation, and Q&A scenarios, including generating descriptions of charts. Datasite [CBYE19] and DataShot [WSZ*20] for instance offer basic statistics or fact sheets using AI models.

Prior attempts at automating description generation for data visualizations include tools like Chartmaster [ZT15] and VoxLens [SWM*22b], which use templates, limiting the depth of information. While some AI models like Chart-to-Text [OH20] and that of Kim et al. [KM18] propose more complex descriptions, they have limitations in chart types or may contain inaccuracies.

VisText [TBS23] authors created a comprehensive dataset consisting of 12,441 crowd-sourced captions specifically designed for

training models to produce detailed and accurate chart descriptions. By assessing the quality of descriptions produced by a model trained on this dataset, the authors also revealed that half of the descriptions were accurate, while the other half contained errors.

Despite efforts to improve accuracy over time, it is acknowledged that similar errors may occur when using other LLMs for generating chart descriptions in various tools, including Generative Pre-training Transformer (GPT). Chat2Vis [MS23] authors proposed a solution that converts natural language directly into code for appropriate visualizations. Even though the scope is the inverse of ours, they showcased the power of LLMs, such as GPT, in supporting DV experts. This breakthrough further motivated the usage of these models to generate descriptions for data visualizations.

DataTales [SS23] also uses GPT to automatically generate chart descriptions. These summaries are chart-agnostic and encompass information up to levels 3 and 4 [LS21]. Nevertheless, certain limitations are observed particularly concerning screen-reader accessibility. Not having an explicit indication that descriptions are generated using an LLM, the absence of a character or sentence limit as well as the lack of level 2 information are some of the ways in which it can be distinguished from AutoVizuA11y.

3. AutoVizuA11y

In this section we present AutoVizuA11y, a tool that automatically adds accessibility features to charts and underlying data (figure 2) by using the information provided by the DV expert during the chart creation process. It supports keyboard navigation, outputs insightful labels and automatic descriptions, and enables multiple shortcuts with various outcomes that enhance navigation and information extraction. To the best of our knowledge, AutoVizuA11y is the first tool to combine all these accessibility features. The tool is a React component — it accepts a set of properties and adds the previously mentioned features to the underlying data visualization SVG elements.

The tool was created through a user-centered design process, with contributions from past research and from a set of exploratory interviews performed with five DV experts and with five SR users.

3.1. Design Requirements

In order to better understand the needs of users, we remotely conducted semi-structured interviews with five DV experts and five SR users. While the DV experts were known peers, the SR users were recruited through known contacts and snowball sampling [CP20]. The last group was awarded a compensation of 25€/$25 gift card.

The findings from the interviews, aligned with previous research on the topic [ZLL*22, JMK*21, EBM22], were fundamental to define a set of requirements that AutoVizuA11y should follow.

The interviews with DV experts mostly focused on trying to find reasons to justify chart inaccessibility. They mostly attributed it to a lack of knowledge and experience in accessibility, as well as insufficient time and tools to support them in this task. Following this contact with DV experts, it was defined that AutoVizuA11y should:

- **Comply with the most relevant accessibility guidelines**, such as WCAG 2.1 and Chartability;

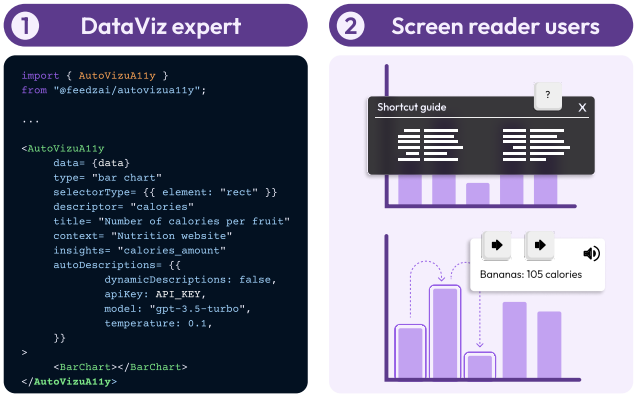

Figure 2: *Diagram demonstrating the two key user groups of AutoVizuA11y: (1) The DV expert imports the tool, uses it in a chart, and provides the properties with the appropriate information. (2) The SR users explore the now accessible chart using a keyboard and screen reader of choice.*

- **Be a package** easily imported and implemented with little intervention from DV experts;
- **Allow to override the automatically generated features**, like descriptions, when needed;

The SR users reported feeling their ability to gather insights is limited because data visualizations are not accessible to them, which leaves them out of important conversations and dependent on sighted peers. This is due to issues like charts being invisible to screen reader software, complexity of the data, clashing of shortcuts, and reliability on table alternatives. Given these insights, AutoVizuA11y should fulfill the following requirements for SR users:

- Make visualization features **navigable and informative**;
- **Warn** SR users **when changes are made** in the visualization;
- **Provide multiple accessible ways** of navigating and understanding the chart;
- **Summarize** the main content of the visualization;
- **Minimize the impact of possible errors and limitations** that may occur from the usage of **LLMs**;
- **Support multiple screen readers**, such as (VoiceOver [App05], JAWS [Sci95], NVDA [Cur06]);
- **Avoid overwhelming** users with information.

3.2. Usage and Integration

AutoVizuA11y is an open-source React library available to any DV expert. It can be installed via npm or used directly by cloning the repository.

AutoVizuA11y works by wrapping each individual chart component. The DV expert must provide a set of properties with basic information regarding the chart and data. These properties enable AutoVizuA11y to correctly identify and label each data element, use the correct values for insights calculation and generate a representative chart description. Some of these are required, like the *data*, *insights*, and *title* while others can be optionally filled (e.g. *descriptor*, *multiSeries*).

When using AutoVizuA11y, the DV expert must choose between manual or automatic data descriptions, selected through the *manualDescriptions* and *autoDescriptions* properties. Choosing one is mandatory. For manual descriptions, both shorter and longer descriptions must be provided, while automatic descriptions only require an OpenAI API key (other model properties are optional).

The current set of properties can be seen in our tool's repository on `https://github.com/feedzai/AutoVizuA11y;`.

3.3. Shortcut Guide

To avoid having the user memorize an overwhelming amount of information, we created a shortcut guide, accessible while focused on either an AutoVizuA11y chart or its underlying elements. We provide the complete list of shortcuts in our paper's supplemental material.

Upon initially focusing an AutoVizuA11y chart, the user is presented with information detailing the tool's functionality and is directed to the shortcut guide. It comprises a list of the 21 available shortcuts. The guide is accessed with ? and closed either with ? or Esc.

While defining the shortcut keys, we considered suggestions provided by SR users during initial interviews, the grouping of shortcuts with similar outputs in the same keyboard area, as well as common meanings from other similar features (e.g. ? is commonly used in Google apps to access a list of shortcuts). Furthermore, to prevent conflicts between AutoVizuA11y shortcuts and those already in use by common browsers and SRs, we avoided any key combinations employed by Chrome, Edge, Firefox, and Safari, as well as those utilized by JAWS, NVDA, and VoiceOver.

3.4. Automated Description Generation

Users of AutoVizuA11y can access two descriptions of a chart. They vary in length and amount of details shared about the data. The descriptions are generated with OpenAI's GPT API. After a series of iterations and considerable prompt engineering to reduce errors, we reached a solution capable of generating textual descriptions that appear to be accurate and informative to SR users. The information present in these two automatic descriptions, together with the title and chart type, are a mix of Levels 1 through 3 [LS21], with context (level 4) varying depending on the information passed in the tool properties.

We initiated the engineering of descriptions by considering both a visual and a data summary of the chart. The visual aspect was dismissed based on feedback from SR users, who considered that information secondary. Although we retained the concept of a data description, the structure of the information underwent significant changes. Initially, the model received information about the axis, chart orientation, maximum, and minimum values. However, it was found that the first two did not contribute additional information to the data summary. The last two insights were easily identified by the model without any instances of hallucination — unlike the average. The current version of the prompt, uses the following variables provided by the DV expert: `context`, `title`, `average` of numerical values, and the actual `data`.

Identity errors, defined in VisText [TBS23] as when a statement incorrectly reports an independent variable for a given trend, were recurrent in the initial iterations. As an example, we were testing a chart with an encoding "days of the week" and, given the structure of the raw data, the assumed starting day kept altering between Monday and Sunday. The solution found became a meaningful set of keys and values (passed in the *data* prop) that create a direct match between encodings.

The longer description is the first being generated and does not have any length boundaries. It can be announced using the Alt + B shortcut. Once the longer description is generated, it is sent back to the model so that a shorter, size constrained, one can be created. This shorter description is the one heard by default, announced when focusing a chart, but can also be announced using the Alt + S key.

The final prompts, sent to an OpenAI's model are:

- Longer description prompt: "Knowing that the chart below is from a [context] and the data represents [title] with an average of [average], make a description (do not use abbreviations) with the trends in the data, starting with the conclusion: [data]";
- Shorter description prompt: "Summarize (in less than 60 words) the following: [longer_description]";

Furthermore, the chart title and type are appended before each model response. As these descriptions are generated automatically and could potentially contain errors, an "Automatic Description:" message is announced by the screen reader between the type and the model's output. This message is hidden if overwritten by the chart's creator using *manualDescriptions*.

Below is an example of longer and shorter descriptions generated by AutoVizuA11y for the first chart in a set of example visualizations created to showcase the tool's functionality. This chart can be viewed in the supplemental materials or explored via `https://diogorduarte.github.io/autovizua11y-examples/`. After focusing the chart, the SR announces the title, "China and India are the most populous countries in the world," accompanied by its type, "bar chart". Subsequently, the cue "Automatic description:" signals the beginning of the description, succeeded by one of the following:

- (**Longer**) The data from the chart represents the 10 most populated countries according to UN statistics from 2022. The two most populated countries are China and India, with China having a population of 1,425.89 million and India having a population of 1,417.17 million. Based on this data, it can be concluded that China and India are the most populous countries in the world. They have a combined population of approximately 2,843.06 million, which is significantly higher than the populations of the other countries on the list. The trend in the data shows that the population of China and India is considerably larger than the populations of the other countries. The difference between the populations of China and India and the populations of the other countries is quite significant, with the next most populous country, the USA, having a population of 338.29 million, which is less than a quarter of the population of China and India. Overall, the data highlights the immense population size of China and India, indicating they have a significant impact on

global demographics and population trends.

- (**Shorter**) China and India are the most populous countries in the world, with a combined population of approximately 2,843.06 million. The population of China and India is considerably larger than the populations of other countries, with the next most populous country, the USA, having a population less than a quarter of China and India's. This data emphasizes the significant impact of China and India on global demographics and population trends.

3.5. Navigation and Labeling

We drew inspiration from Olli's work [BZS22] to design the navigation structure of AutoVizuA11y. Our tool allows SR users to navigate between three levels of information using just the keyboard: chart, data elements, and data series (if more than one is present).

Keyboard navigation is set through a series of interventions in the chart. Upon loading the page, to prevent conflicts with the added features, the tool removes any legacy *tabIndexes* and *ARIA* attributes inside the charts. After that, it identifies the DOM element containing the chart and makes it navigable by adding a `tabIndex="0"`. We refer to the navigation between AutoVizuA11y charts as ***Chart level*** navigation. The title, chart type, and description are added to each one afterward.

The navigation between charts and data elements is done horizontally using the ← and → keys. This prevents the user from navigating outside the chart, as would occur with the ⇥ key. Alternatively, it is still possible to use the ⇥ key to move forward and ⇧+⇥ to move backwards.

To navigate to the data elements, the user should press the ↓ while focused on a chart. When they do so, AutoVizuA11y wipes the `tabIndex` attributes from all AutoVizuA11y charts and adds it to each data element in the focused chart — the tool identifies the DOM elements by their `type` or `className`, chosen by the DV expert. Once that is done, the focus is set to the first data element of the chart. We refer the set of data elements as ***Data level***. If ↑ is pressed while focused on a data element, the process is reversed.

SR users can also quickly move inside each chart using shortcuts independently of the data size. It is possible to jump to the first (Alt+Q or Home) and last data elements (Alt+W or End), avoiding the need to go through every point to reach either side.

It is also possible to define the number of points to jump at a time, the default value being `1`. This can be done by pressing Alt+X, which then prompts the user to *"Enter a number above 0"*. This number is defined per chart, does not influence others, and is kept even if the focus changes to another chart. It is also possible to define this number using the + to add one to the number of points to be jumped and - to subtract one.

Finally, in case there are multiple series of data points in a chart (for instance, a line chart with different lines representing different entities), the SR users can alternate between them using Alt+M while on the *Data level*. When changing between series, the index of the focused data element will remain the same, meaning the focus will change from, for example, the third element of series A to the third element of series B.

3.6. Insights

We previously discussed how SR users find statistical insights necessary to better understand a chart. AutoVizuA11y makes this information more easily accessible. Current approaches involve copying the datasets to spreadsheet software, and performing mental calculations while navigating between elements. AutoVizuA11y, similarly to tools like Voxlens [SWM*22b], determines these insights based on the data provided and shares them with users after the right key combinations are pressed. In either *Chart level* or *Data level*, the user can request the minimum (Alt+J), average (Alt+K), and maximum (Alt+L) values. All three shortcuts make the screen reader announce, using `aria-live="assertive"` (automatically set by our tool), the following: *"The [insight] is [value]"*.

While in the *Data level*, it is also possible to compare the focused data element against the same statistical insights. The SR announces *"The value is [difference] [X] the [insight]"* where X can be *"below"*, *"above"* and *"the same as"*.

A user can find the position of a data point relative to others in the chart by pressing Alt+Z, with AutoVizuA11y announcing *"This is the {maximum OR minimum OR median} value"*, or *"This is the [ordinal_numeral] {highest OR lowest} value"*.

In the case of a multi-series chart, requesting any of the previous shortcuts associated with insights will yield information regarding all data elements, not limiting the insight to that particular series.

4. Usability study

To assess the performance of the tool, we conducted a remote study using an interface with charts created with AutoVizuA11y. The participants had the opportunity to freely explore the interface before completing a set of eight tasks. During the session we recorded time, success rate, and method of completion for each task. The participants gave their opinions on the charts both verbally and through a System Usability Scale (SUS) questionnaire [Bro95]. The complete set of tasks is provided in the supplemental material.

4.1. Participants

We recruited 15 participants with ages ranging between 23 and 51 years old (`M = 39.33`, `SD = 8.52`). Of those, 11 identified as men and four as women. The sole prerequisite for participation in the study was having daily experience with screen readers. The least experienced was using the technology for the past seven months and the most versed was using it for 30 years (`M = 19.37`, `SD = 7.33`). Two were from the United States while 13 were from Portugal. Thirteen participants reported total blindness, while the remaining two indicated having low vision. None of the participants were born blind, and the earliest that someone became a SR user was at seven years of age. All of them used a computer screen reader daily and, in the sessions, 12 used NVDA while two used Voiceover and one JAWS. The recruitment was done through peers of previous contacts, the portuguese Associação dos Cegos e Amblíopes de Portugal (ACAPO) [dCeAdP89] association, and the "blindsurveys" subreddit. Participants were compensated with a 25€/$25 gift card.

4.2. Procedure and Design

For the participants' comfort, we opted to conduct the studies remotely, using either Zoom or Google Meet, and also allowed them to use any browser and SR of their choice.

Each participant was tasked with exploring an interface with ten charts, rendered side to side, created using AutoVizuA11y (figure 3). They included four horizontal single-stacked bar charts, one heatmap, one vertical bar chart, one boxplot, two horizontal bar charts, and one list of counters. The interface's scenario revolved around a fictional bank account dashboard, and the layout, as well as the chart types, were based on a real-world scenario.

The interface was hosted online and shared with the participant. Each session was recorded with permission of the participants. The users only shared their screen while exploring the charts.

Metrics collected include time-per-task, success rate and method of completion (shortcuts, navigation and descriptions). The eight tasks spanned across different charts in the interface. Participants were asked, for example, to count the total number of values; find the minimum, maximum, and other specific values; calculate the average; and compare a value against others.

The tasks were presented to all participants in a fixed order. This way, we ensured that all participants not only had a similar experience but also got to explore all chart types and AutoVizuA11y features. Additionally, we ensured that every AutoVizuA11y shortcut could be used to solve, at least, one task.

We opted to have the descriptions being automatically generated during the session, acknowledging the potential for errors (as defined in VisText [TBS23]). Nevertheless, we were confident in this approach based on the consistently accurate results observed with the charts tested prior to the sessions. Additionally, since we could hear the output from the SR in the shared screen, the moderator could promptly identify any errors in the descriptions. No errors were identified during the sessions.

After all tasks were completed, participants gave their verbal feedback regarding the interface through an unstructured interview and completed a SUS questionnaire [Bro95].

4.3. Results

Below are the results and takeaways from the usability test conducted in which 15 SR users completed eight tasks (T) in one interface with charts built using AutoVizuA11y. In total, 120 observations were collected.

Only one time measurement was discarded since the participant gave up on completing T4. An interquartile range analysis was applied to the time-per-task data, resulting in nine outliers (out of 119 recorded times) with a maximum of two per participant. All these outliers were skewed towards the higher time values. Given the low number of outliers and since no patterns were found in a participant, all of them were considered for analysis.

Regarding success rate, all 120 recorded tasks were considered and the single one that was abandoned (T4) by a participant was counted as a wrong answer.

4.3.1. Time-per-task

We recorded the time each participant took in each task (fig. 4). The average time taken per task in the interface with AutoVizuA11y charts was 66 seconds (`SD = 59.96s`). Among the tasks, T4 (`M = 116.64s; SD = 79.25s`) — that required users to identify the difference between a data point and the average —, stood out for its longer completion time. This could be attributed to the task's complexity, involving multiple pieces of information required to achieve a correct answer (value of a data point, average of the data, and the difference between them).

The highest recorded time was 320 seconds, observed in T4, where a participant forgot they could navigate to the data elements. The need to explore the interface during tasks, such as consulting the shortcuts guide, likely contributed to extended completion times for certain tasks.

In contrast, T1 (`M = 34.43s; SD = 33.13s`), where participants had to identify the total number of values in a chart, had the lowest average completion time (`M=51.36s; SD=55.57s`). Interestingly, this was also the task where the fastest completion time (four seconds) was recorded.

Among the methods used in the charts interface, descriptions had, on average, the shortest times (43 seconds), followed by navigation (75 seconds) and shortcuts (80 seconds).

4.3.2. Success rate

We recorded whether or not the participants successfully completed each task (fig. 4). The success rate across tasks indicates that SR users consistently perform well in tasks in charts created with AutoVizuA11y, achieving success in 107 out of 120 tasks, approximately 89%.

Task 3, requiring participants to find the minimum value among a small number of points, stood out with a 100% success rate. Tasks 7 and 8 closely followed, each with 14 out of 15 correct answers.

In contrast, Task 1 had the lowest success rate at 80%, possibly due to the learning curve associated with experiencing a new tool.

One user managed to successfully respond to only half of the tasks, while seven participants answered all tasks correctly.

Among the methods used in the charts interface, shortcuts had the highest success rate at 96% (27 correct answers out of 28), followed by navigation with 91% (48 correct answers out of 53). Descriptions achieved an 84% success rate (32 out of 38).

4.4. Participant's feedback

After interacting with the charts, participants (P) were asked to provide additional feedback through an unstructured interview. This portion of the sessions was later transcribed and coded by one of the authors. The results of the thematic analysis [CB16] are presented below:

4.4.1. Overall positive feedback

Thirteen participants verbally expressed their overall satisfaction with the charts built with AutoVizuA11y, highlighting the small

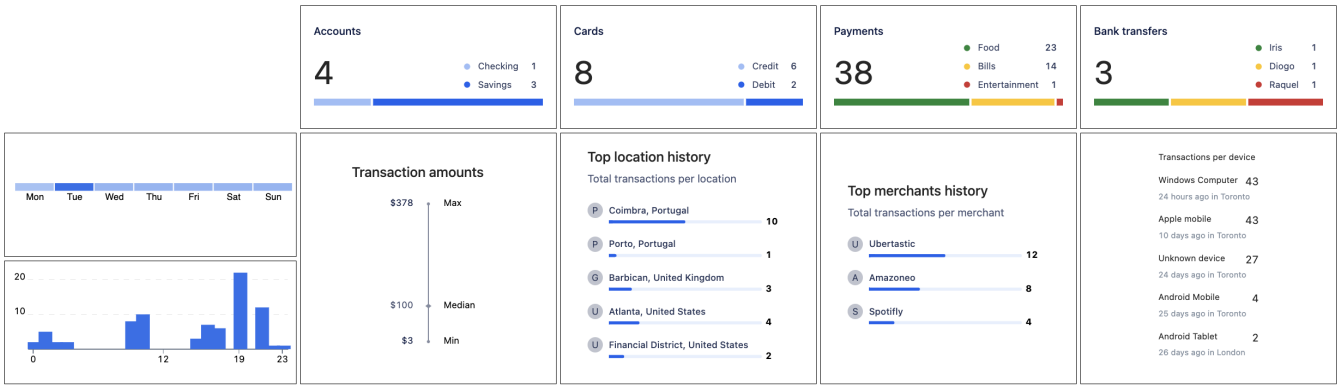

Figure 3: *The interface tested by SR users with ten accessible charts created using AutoVizuA11y.*

learning curve and how included it made them feel. "I am used to working with tables but the data presented in these charts deviates from the tabular format while also offering a more rich and inclusive representation," noted P8. "I never experienced a similar tool (...) It has a very short learning curve (...) I instantly understood the logic behind it" they added. Participants also mentioned the impact AutoVizuA11y would have in other contexts: "The charts really captivated me (...) during high school I could not assess information present in charts and having them finally accessible is something that was new for me and aroused my interest," shared P3. Other participants commented on the usage of AutoVizuA11y in their daily routines. "In my banking app I get lost in their charts and these ones seem much more intuitive," remarked P5. However, two participants had some questions about the tool's usefulness. P7

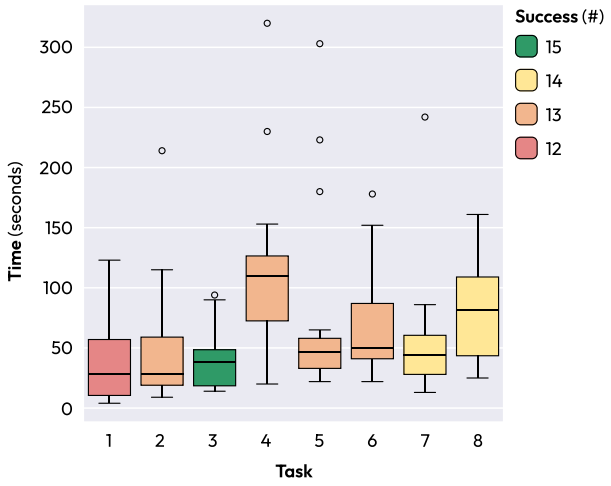

Figure 4: *Quantitative results from a usability test involving 15 SR users who completed eight tasks in AutoVizuA11y charts. A series of boxplots, generated using the interquartile range method, illustrates the time taken per task. Each box is colored based on the number of successful answers provided for each task.*

and P11 challenged the amount of shortcuts and claimed that they would still prefer to explore data through a table.

4.4.2. Navigation

Most participants understood the navigation between charts and data elements. "I found it interesting that the navigation just used the arrow keys," remarked P2. P14 shared a similar sentiment: "The navigation with the arrow keys was simple enough for me".

Users quickly navigated inside the charts using the arrow keys. While one noted dependence on user concentration, additional feedback highlighted it as an outlier. It was also pointed out that the navigation inside the shortcut guide did not match their expectations. P5 and P6 agreed that it would be "good to have the same navigation using the arrow keys inside the guide", which highlights the need for consistency across the entire tool.

4.4.3. Shortcuts

Participants who experimented with the shortcuts found them very useful. P6 stated that, without the shortcuts, "would need to either do the calculations mentally or use a calculator that consumes more time". P10 agreed that the feature "greatly speeds up the process".

Despite receiving positive feedback, P1 and P4 expressed concerns about insufficient time to explore all shortcuts. "I knew there were other shortcuts but in the moment I did not remember them (...) I know that next time I will use them once they get more familiar," explained P1. P4 shared a similar sentiment, stating, "I found the navigation to be good but had some issues with the shortcuts (...) for context, I use the SR daily and only remember a few (shortcuts), but having the guide available helps greatly."

4.4.4. Descriptions

Participants praised the ability to change between the two types of descriptions and the insights they brought to them. "I liked the long and short descriptions and that you could rapidly change between one another," remarked P9. P15 echoed this sentiment, stating, "The descriptions are very useful (...) they provide insights similar to the ones I believe a sighted person would gather just by quickly looking at the chart." Additionally, P9 expressed interest

in having each chart type described: "For the sake of curiosity, I would like to also hear a description of the shape of the chart".

4.4.5. Obstacles and suggestions

Participants identified some barriers that hindered the smooth usage of the tool. Alongside the absence of arrow navigation within the shortcut guide, several participants highlighted the necessity to manually activate the focus mode from NVDA: (P14) "An NVDA user would eventually get it, but ideally, they would not need to turn on focus mode." Additionally, P9 mentioned Sonification as a potential addition to the existing features, despite personal preference: (P9) "I know people like to hear sounds (sonification) for the shape of the data, but I do not really like that. I prefer it the way is being described now." Lastly, P10 emphasized the importance of supporting other assistive technologies beyond screen readers: (P10) "I would like to test the tool using a braille line".

4.5. System Usability Scale Score

After providing verbal feedback, users evaluated the interface with the charts built using AutoVizuA11y through a SUS questionnaire [Bro95]. The final SUS score was `83.5`, considered "Excellent" [BKM08]. The average responses to the questionnaire indicate that participants perceived the interface's usability very positively. They found the tool easy to use, well-integrated, and expressed confidence in using it without requiring much support. Overall results suggest a highly favorable user experience with the tool.

That said, there were two users that rated the interface considerably lower than the rest: 37.5 and 42.5 respectively. One of them (P7) abandoned a task and unsuccessfully answered half of the tasks. The other (P11) disclosed a learning disability after the session, justifying the difficulty in grasping our tool's new interaction method within the limited session time. If these two participants were to be excluded from the final SUS score calculation, the tool's score would be `90.2`.

5. Data Visualization experts feedback

After releasing AutoVizuA11y as an open-source tool, we engaged with two DV experts (E), one of which participated in the interviews, to test a beta version. Their task was to incorporate the tool into a project of their choice and share feedback on its functionality. After using the tool, both DV experts (E) expressed their appreciation for its development and acknowledged the effort invested in making charts and visualizations more accessible. E1 stated "Thank you very much for making this project open source so we can benefit from it too." The other expert, E2, emphasized the tool's role in lowering barriers to accessibility, noting, "I really think this (AutoVizuA11y) lowers the barrier to making charts and visualizations more accessible for everyone."

When asked about the likelihood of recommending AutoVizuA11y to others, both DV experts indicated that it was likely. They also pointed to areas for improvement, particularly from the perspective of web DV experts, that we summarize below.

The DV experts encountered challenges when using AutoVizuA11y in frameworks other than React. They emphasized the necessity for clearer documentation, especially regarding the expected data format — this was already changed in the most recent version of the tool from object to an array of objects, enabling support for more than two chart encodings and aligning closely with the format used in many visualization libraries.

Language support beyond English emerged as a key requirement. DV experts also advocated for a more modular "description backend" to seamlessly integrate with other services/APIs.

Although the tool was tested with various visualization libraries, the examples on the website exclusively featured charts built with visx. In response to this feedback, we will add examples with more visualization libraries to highlight the tool's versatility.

Both DV experts stressed the importance of typing syntax support, crucial for broader compatibility with TypeScript projects and aiding in debugging. This aligns with another raised concern — lack of error handling.

The feedback, though from a small sample, highlights that AutoVizuA11y effectively addresses an existing gap in DV expert's process. As accessibility for SR users begins with DV experts, and considering the feedback obtained from those contacted, it is prudent to enhance the tool's compatibility with various other workflows in the future.

6. Discussion

Feedback from SR users and DV experts emphasizes the significant potential of AutoVizuA11y as a tool that enhances data visualization accessibility.

SR users who explored visualizations created with AutoVizuA11y showed a strong interest in the tool's user experience. Participants carefully listened to the entire description of each chart at least once, allowing them to have a quick overview of the data.

Participants found the data navigation and shortcuts useful, and highlighted that the shortcut guide was essential. Concerns were raised about the tool's clarity without a moderator, which prompted the addition of a short tutorial (skippable) at the beginning of the shortcut guide.

Although only two DV experts used the tool, both recognized the value of AutoVizuA11y and agreed that it streamlines their accessible chart creation process. As the experience of SR users depends on the accessibility incorporated into charts, supporting a wide range of DV expert workflows becomes invaluable. Consequently, we believe it is crucial to support as many chart types, visualization libraries, or technologies as possible when building tools of similar scope.

We are confident that the current iteration of AutoVizuA11y represents a significant step towards enhancing accessibility in the data visualization space. AutoVizuA11y enables data visualizations to be explored and enjoyed by a larger group of people, some of which have been previously excluded from doing so. This is manifested in the praise it received from the majority of SR users, who suggested its application in various real-world scenarios, and from DV experts that see value in a tool like AutoVizuA11y.

6.1. Limitations and Future Work

6.1.1. Limitations

There is one structural limitation related to the tool composition. The prompt uses raw data, but OpenAI's token size limit restricts data usage. Possible solutions include using aggregated data or doing multiple API calls.

In regards to the usability study, due to session time constraints, smaller datasets were used. Even though AutoVizuA11y has shortcuts that quicken the keyboard navigation, we believe a larger dataset could have potentially affected the duration of the sessions.

Given the unique features and comprehensiveness of AutoVizuA11y, we made the decision not to compare it against other tools or visualization libraries. This results from us being unable to identify any existing tool offering a comparable array of accessibility features. Moreover, introducing multiple new approaches to participants within the time frame of the sessions would have likely constraint the insights gathered. However, we acknowledge that this decision represents a limitation of the study. Moving forward, we would like to conduct a comparative evaluation of AutoVizuA11y, potentially against a group of tools simultaneously.

Due to time constraints, it was only possible to show the tool to two DV experts. Both stated the tool was easy to understand and implement. Regardless, our plan is to further test AutoVizuA11y with a wider range of DV experts. This will not only help us understand the intuitiveness of the tool but also enable us to identify and address any potential shortcomings that arise.

6.1.2. Future work

We intend to add previously identified features in upcoming iterations of the tool, like the ability to search for specific data points within the interface. This would enhance the users ability to access specific information on demand.

Considering the strong reliance by SR users on tables as chart alternatives, we believe its crucial to expand AutoVizuA11y to support tables and make the necessary adaptations to the tool.

Additionally, in cases where more than one chart is available, providing an overall summary of all charts' descriptions within an interface could offer a comprehensive understanding of the available visualizations in case they share a context. Similarly, enabling users to set preferences for specific chart types across multiple interfaces would personalize their experience and enhance usability. Participants in the usability test also suggested incorporating sonification, automatically switch to "Focus" mode for JAWS and NVDA users, and enhancing navigation within the shortcut guide.

As mentioned in the discussion of the results (section 6), it can make sense to explain the visual aspects of each chart beyond the existing description. This enhancement could involve detailing the colors of each element or explaining the chart type, similar to the approach taken by Kim et al. [KKK23]. By offering this information as an option, we avoid overwhelming those who may not consider it essential.

Finally, although it was not an initial goal of the project, non-visually impaired users asked how they could also make use of AutoVizuA11y generated descriptions, data insights, and keyboard navigation. This opens a potential new line of research on how to design an interface that can provide those outputs even without a screen reader.

7. Conclusion

We introduced AutoVizuA11y, a tool designed to automate the creation of accessible data visualizations for SR users. Our iterative research [ND86] and validation process involving SR users enabled the development of meaningful features, while also streamlining the necessary knowledge required from a creator's standpoint.

AutoVizuA11y automatically improves the accessibility of charts by adding shortcuts that offer rapid answers without the need for calculations, human-like descriptions of the data, and keyboard navigation that enables on-demand access to specific information. It also consolidates previously introduced features into a single tool and enhances them, ensuring that DV experts do not need to use multiple solutions to ensure extensive accessibility for SR users.

Fifteen SR users tested an interface containing accessible charts built with the assistance of AutoVizuA11y. While the usage of our tool had an expected learning curve, the time-per-task, success rate, and verbal feedback received confirmed AutoVizuA11y's potential. The participants also completed a SUS questionnaire pertaining to the interface tested, granting it a score of 83.5 ("Excellent" [BKM08]).

Despite the common recommendation in the literature of having a table alternative for each chart present on the web, we believe that our tool proves otherwise. The majority of participants agreed that if the charts alone provide similar insights to those retrieved by AutoVizuA11y, there is no need to resort to a table.

This work also proved that it is possible to automatically generate data visualization descriptions of higher complexity using LLMs, without the need for a person to manually analyze the key insights of a given visualization.

While the tool needs details about the data to operate, it does not require extensive accessibility knowledge. AutoVizuA11y's implementation is less demanding as the needed information is typically already available during the creation of the chart.

Finally, the majority of participants brought light to the impact our tool could have in their lives if available to a broader public. Because of this, the tool is available via open-source, and can be found at https://github.com/feedzai/AutoVizuA11y.

8. Acknowledgements

This was a collective work supported by both Feedzai and by FCT through the LASIGE Research Unit, ref. UIDB/00408/2020 (https://doi.org/10.54499/UIDB/00408/2020) and ref. UIDP/00408/2020 (https://doi.org/10.54499/UIDP/00408/2020). We extend our gratitude to the ACAPO for their support and assistance in the recruitment process and all involved participants. We also express our appreciation to the reviewers for their valuable comments and suggestions.

References

[AFP24] AFP: Screen readers, 2024. URL: https://www.afb.org/blindness-and-low-vision/using-technology/assistive-technology-products/screen-readers.

[Air17] AIRBNB: visx homepage, 2017. URL: https://airbnb.io/visx.

[App05] APPLE: Voiceover user guide for mac, 2005. URL: https://support.apple.com/guide/voiceover/welcome/mac.

[BBK*15] BORKIN M. A., BYLINSKII Z., KIM N. W., BAINBRIDGE C. M., YEH C. S., BORKIN D., PFISTER H., MEMBER S., OLIVA A.: Beyond memorability: Visualization recognition and recall. URL: http://massvis.mit.edu.

[BKM08] BANGOR A., KORTUM P. T., MILLER J. T.: An empirical evaluation of the system usability scale. *International Journal of Human–Computer Interaction 24*, 6 (2008), 574–594. URL: https://doi.org/10.1080/10447310802205776, doi:10.1080/10447310802205776.

[Bli20] Causes of blindness and vision impairment in 2020 and trends over 30 years, and prevalence of avoidable blindness in relation to vision 2020: the right to sight: an analysis for the global burden of disease study. URL: http://www.anglia.ac.uk/verigbd, doi:10.1016/S2214-109X(20)30489-7.

[Bro95] BROOKE J.: Sus: A quick and dirty usability scale. *Usability Eval. Ind. 189* (11 1995).

[BZS22] BLANCO M., ZONG J., SATYANARAYAN A.: Olli: An extensible visualization library for screen reader accessibility. URL: http://vis.csail.mit.edu/pubs/olli/, doi:10.1109/VIS47514.2020.00033.

[CB16] CLARKE V., BRAUN V.: Thematic analysis. *The Journal of Positive Psychology 12*, 3 (Dec. 2016), 297–298. URL: http://dx.doi.org/10.1080/17439760.2016.1262613, doi:10.1080/17439760.2016.1262613.

[CBYE19] CUI Z., BADAM S. K., YALÇIN M. A., ELMQVIST N.: Datasite: Proactive visual data exploration with computation of insight-based recommendations. *Information Visualization 18* (4 2019), 251–267. URL: https://journals.sagepub.com/doi/full/10.1177/1473871618806555, doi:10.1177/1473871618806555/ASSET/IMAGES/LARGE/10.1177_1473871618806555-FIG2.JPEG.

[CP20] CHARLIE PARKER S. S. . A. G.:, 2020. URL: http://dx.doi.org/10.4135/9781526421036831710, doi:10.4135/9781526421036831710.

[Cur06] CURRAN M.: https://www.nvaccess.org/, 2006. URL: https://www.nvaccess.org/.

[dCeAdP89] DOS CEGOS E AMBLÍOPES DE PORTUGAL A.: Acapo homepage, 1989. URL: https://www.acapo.pt/.

[Dow13] DOWNIE N.: Chart.js homepage, 2013. URL: https://www.chartjs.org/.

[EBM22] ELAVSKY F., BENNETT C., MORITZ D.: How accessible is my visualization? evaluating visualization accessibility with chartability. *Computer Graphics Forum 41* (2022), 57–70. URL: https://onlinelibrary.wiley.com/doi/10.1111/cgf.14522, doi:10.1111/cgf.14522.

[EMW19] ENGEL C., MÜLLER E. F., WEBER G.: Svgplott: An accessible tool to generate highly adaptable, accessible audio-tactile charts for and from blind and visually impaired people. URL: https://doi.org/10.1145/3316782.3316793, doi:10.1145/3316782.3316793.

[ENM23] ELAVSKY F., NADOLSKIS L., MORITZ D.: Data Navigator: an accessibility-centered data navigation toolkit. *IEEE VIS* (2023).

[Hig09] HIGHSOFT: Highcharts homepage, 2009. URL: https://www.highcharts.com.

[JMK*21] JUNG C., MEHTA S., KULKARNI A., ZHAO Y., KIM Y.-S.: Communicating visualizations without visuals: Investigation of visualization alternative text for people with visual impairments, 2021. URL: http://arxiv.org/abs/2108.03657.

[KJRK21] KIM N. W., JOYNER S. C., RIEGELHUTH A., KIM Y.: Accessible visualization: Design space, opportunities, and challenges. *Computer Graphics Forum 40*, 3 (June 2021), 173–188. URL: https://doi.org/10.1111/cgf.14298, doi:10.1111/cgf.14298.

[KJRK22] KIM N. W., JOYNER S. C., RIEGELHUTH A., KIM Y.: Visualization accessibility in the wild: Challenges faced by visualization designers. ACM, pp. 1–19. URL: https://dl.acm.org/doi/10.1145/3491102.3517630, doi:10.1145/3491102.3517630.

[KKK23] KIM G., KIM J. ., KIM Y.-S.: "explain what a treemap is": Exploratory investigation of strategies for explaining unfamiliar chart to blind and low vision users;. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23), April 23-28, 2023, Hamburg, Germany 1* (2023). URL: https://doi.org/10.1145/3544548.3581139, doi:10.1145/3544548.3581139.

[KM18] KIM E., MCCOY K. F.: Multi modal deep learning using images and text for information graphic classification. *ASSETS 2018 - Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility* (10 2018), 143–148. URL: https://doi.org/10.1145/3234695.3236357, doi:10.1145/3234695.3236357.

[LS21] LUNDGARD A., SATYANARAYAN A.: Accessible visualization via natural language descriptions: A four-level model of semantic content. URL: http://arxiv.org/abs/2110.04406http://dx.doi.org/10.1109/TVCG.2021.3114770/, doi:10.1109/TVCG.2021.3114770/.

[MB11] MIKE BOSTOCK JASON DAVIES J. H. V. O.: D3.js homepage, 2011. URL: https://d3js.org/.

[MS23] MADDIGAN P., SUSNJAK T.: Chat2vis: Generating data visualisations via natural language using chatgpt, codex and gpt-3 large language models. URL: http://arxiv.org/abs/2302.02094, doi:10.1109/ACCESS.2023.3274199.

[ND86] NORMAN D. A., DRAPER S. W.: *User Centered System Design; New Perspectives on Human-Computer Interaction*. L. Erlbaum Associates Inc., USA, 1986.

[OH20] OBEID J., HOQUE E.: Chart-to-text: Generating natural language descriptions for charts by adapting the transformer model. 15–18.

[SB22] SOUTH L., BORKIN M.: Photosensitive accessibility for interactive data visualizations, 2022. URL: https://osf.io/7uyn9.

[Sci95] SCIENTIFIC F.: Jaws screen reader homepage, 1995. URL: https://www.freedomscientific.com/products/software/jaws/.

[SCWR21] SHARIF A., CHINTALAPATI S. S., WOBBROCK J. O., REINECKE K.: Understanding screen-reader users' experiences with online data visualizations. ACM, pp. 1–16. URL: https://dl.acm.org/doi/10.1145/3441852.3471202, doi:10.1145/3441852.3471202.

[SF18] SHARIF A., FOROURAGHI B.: evographs — a jquery plugin to create web accessible graphs. IEEE, pp. 1–4. http://evoxlabs.org/projects/evographs. URL: http://ieeexplore.ieee.org/document/8319239/, doi:10.1109/CCNC.2018.8319239.

[SPF22] SCHWABISH J., POPKIN S., FENG A.: Centering accessibility in data visualization. *Do No Harm Guide* (2022).

[SS23] SULTANUM N., SRINIVASAN A.: Datatales: Investigating the use of large language models for authoring data-driven articles, 2023. URL: https://arxiv.org/abs/2308.04076, doi:10.48550/ARXIV.2308.04076.

[SWM22a] SHARIF A., WANG O. H., MUONGCHAN A. T.: "what makes sonification user-friendly?" exploring usability and user-friendliness of sonified responses. In *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility* (Oct. 2022), ASSETS '22, ACM. URL: http://dx.doi.org/10.1145/3517428.3550360, doi:10.1145/3517428.3550360.

[SWM*22b] SHARIF A., WANG O. H., MUONGCHAN A. T., REINECKE K., WOBBROCK J. O.: Voxlens: Making online data visualizations accessible with an interactive javascript plug-in. ACM, pp. 1–19. URL: https://dl.acm.org/doi/10.1145/3491102.3517431, doi:10.1145/3491102.3517431.

[TBS23] TANG B. J., BOGGUST A., SATYANARAYAN A.: Vistext: A benchmark for semantically rich chart captioning, 2023. URL: http://vis.csail.mit.edu/pubs/vistext/.

[W3C19] W3C: Wcag 2.1, 2019. URL: https://www.w3.org/WAI/WCAG22/quickref/.

[Web21] WEBAIM: Screen reader user survey 9 results, 2021. URL: ttps://webaim.org/projects/screenreadersurvey9/.

[WPA*21] WU K., PETERSEN E., AHMAD T., BURLINSON D., TANIS S., SZAFIR D. A.: Understanding data accessibility for people with intellectual and developmental disabilities. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2021), CHI '21, Association for Computing Machinery. URL: https://doi.org/10.1145/3411764.3445743, doi:10.1145/3411764.3445743.

[WSZ*20] WANG Y., SUN Z., ZHANG H., CUI W., XU K., MA X., ZHANG D.: Datashot: Automatic generation of fact sheets from tabular data. *IEEE Transactions on Visualization and Computer Graphics 26* (1 2020), 895–905. doi:10.1109/TVCG.2019.2934398.

[WWJK22] WANG Y., WANG R., JUNG C., KIM Y.-S.: What makes web data tables accessible? insights and a tool for rendering accessible tables for people with visual impairments. ACM, pp. 1–20. URL: https://dl.acm.org/doi/10.1145/3491102.3517469, doi:10.1145/3491102.3517469.

[ZLL*22] ZONG J., LEE C., LUNDGARD A., JANG J., HAJAS D., SATYANARAYAN A.: Rich screen reader experiences for accessible data visualization. *Computer Graphics Forum 41* (2022), 15–27. URL: https://onlinelibrary.wiley.com/doi/10.1111/cgf.14519, doi:10.1111/cgf.14519.

[ZT15] ZOU H., TREVIRANUS J.: Chartmaster: a tool for interacting with stock market charts using a screen reader. URL: http://dx.doi.org/10.1145/2700648.2809862, doi:10.1145/2700648.2809862.